THE UNIVERSITY *of* EDINBURGH
informatics

Extreme Computing

BitTorrent and incentive-based overlay networks

BitTorrent

- Today we will focus on BitTorrent

- The technology really has three aspects
  - A standard that BitTorrent client systems follow
  - Some existing clients, e.g., the free Torrent client, PPLive
  - A clever idea: using “tit-for-tat” (incentive) mechanisms to reward good behavior and to punish bad behavior

- This third aspect is especially intriguing

Why is (studying) BitTorrent important?

- An organic, large-scale P2P network
  - That scales according to use
  - Incentive-based: the more you give, the more you get
- Used as a delivery method for multiple media
  - Not only illegally obtained copyrighted material
  - Linux iso's delivery
  - (Legal) Media content distribution

- November 2004: BitTorrent responsible for 35% of all Internet traffic.
- February 2009: P2P networks account for approximately 43% to 70% of all Internet traffic (depending on geographical location)
- January 2012: 150 million active users
  - Monthly users projected to 1 billion
- February 2013: BitTorrent responsible for 3.35% of all worldwide bandwidth
  - More than half of the 6% of total bandwidth dedicated to file sharing

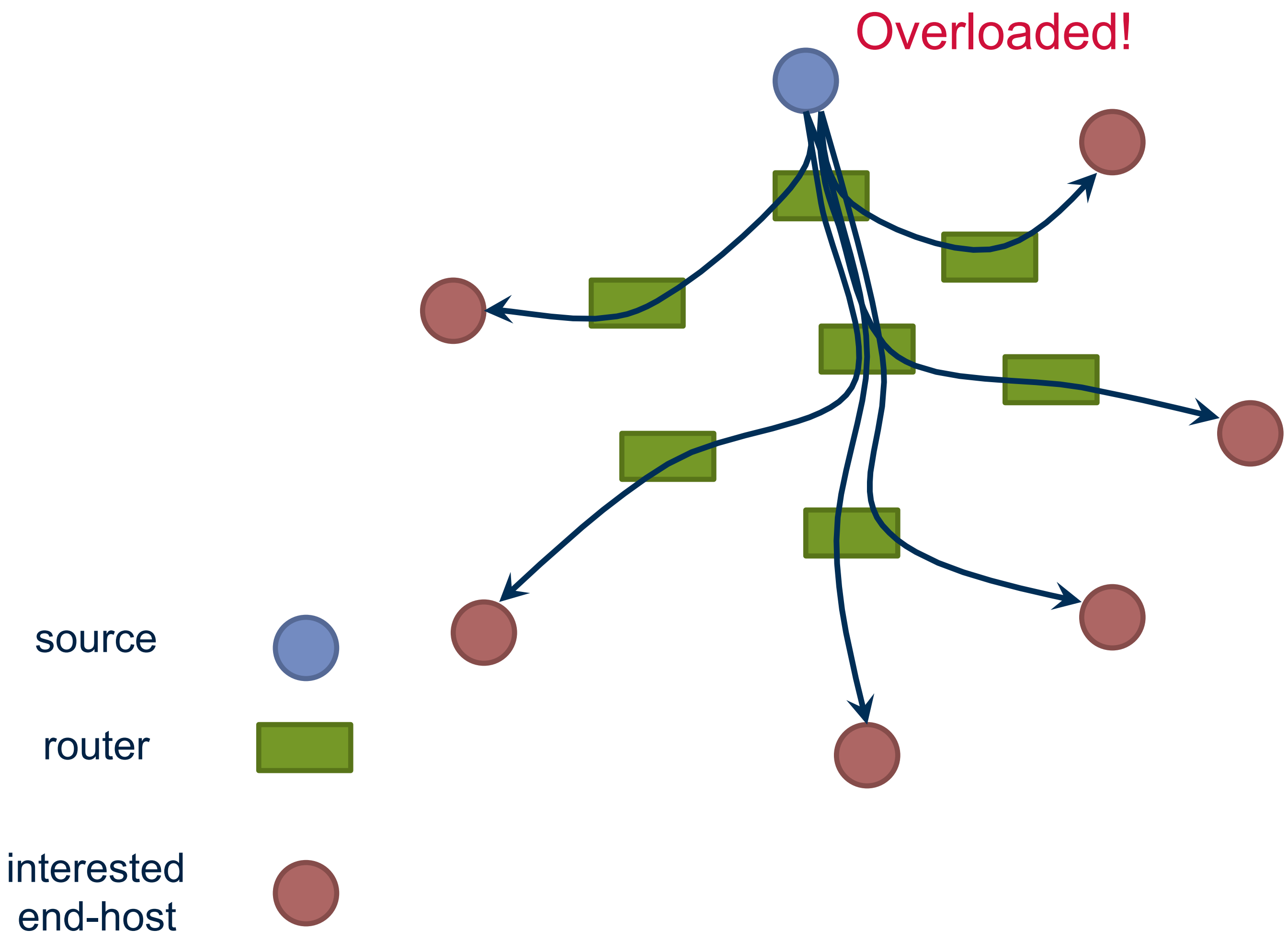
The basic BitTorrent scenario

- Millions want to download the same popular huge files (for free)
  - ISO’s
  - Media (the real example!)
    - And the one that gave BitTorrent a bad rep
- Client-server model fails
  - Single server fails
  - Cannot afford to deploy enough servers
- Why not IP multicast?
  - Not a real option in general WAN settings
    - Not supported by many ISPs
  - Most commonly seen in private data centers
- Alternatives
  - End-host based Multicast
  - BitTorrent
  - Other P2P file-sharing schemes (from prior lectures)

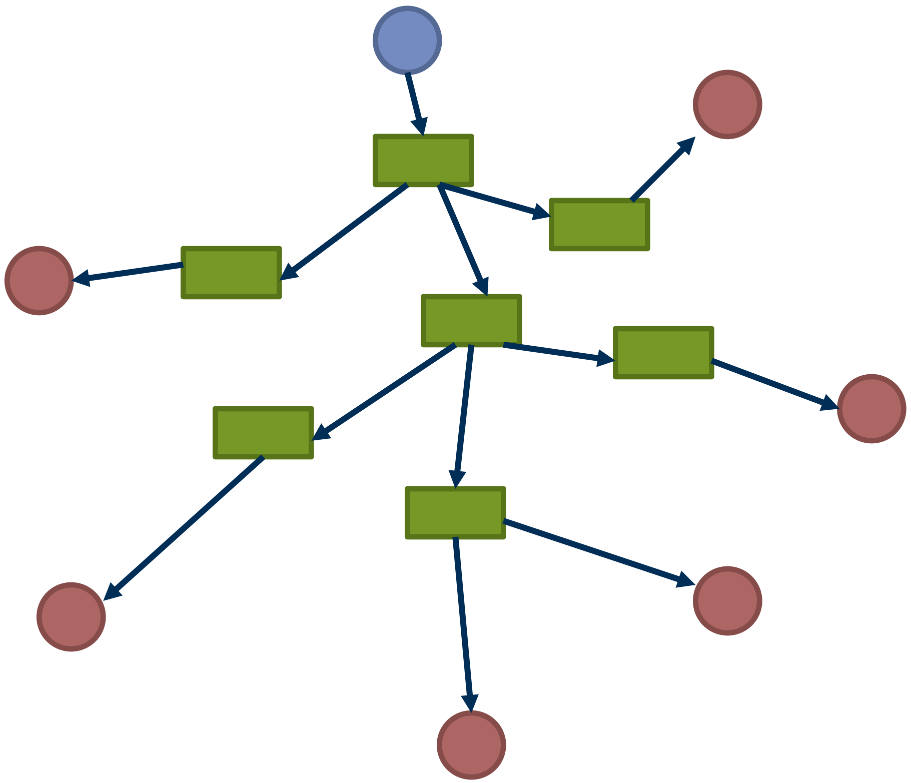
Why not use IP Multicast?

- IP Multicast not a real option in general WAN settings
  - Not supported by many ISPs
  - Most commonly seen in private data centers
- Alternatives
  - End-host based Multicast
  - BitTorrent
  - Other P2P file-sharing schemes (from prior lectures)

Traditional client-server



IP multicast

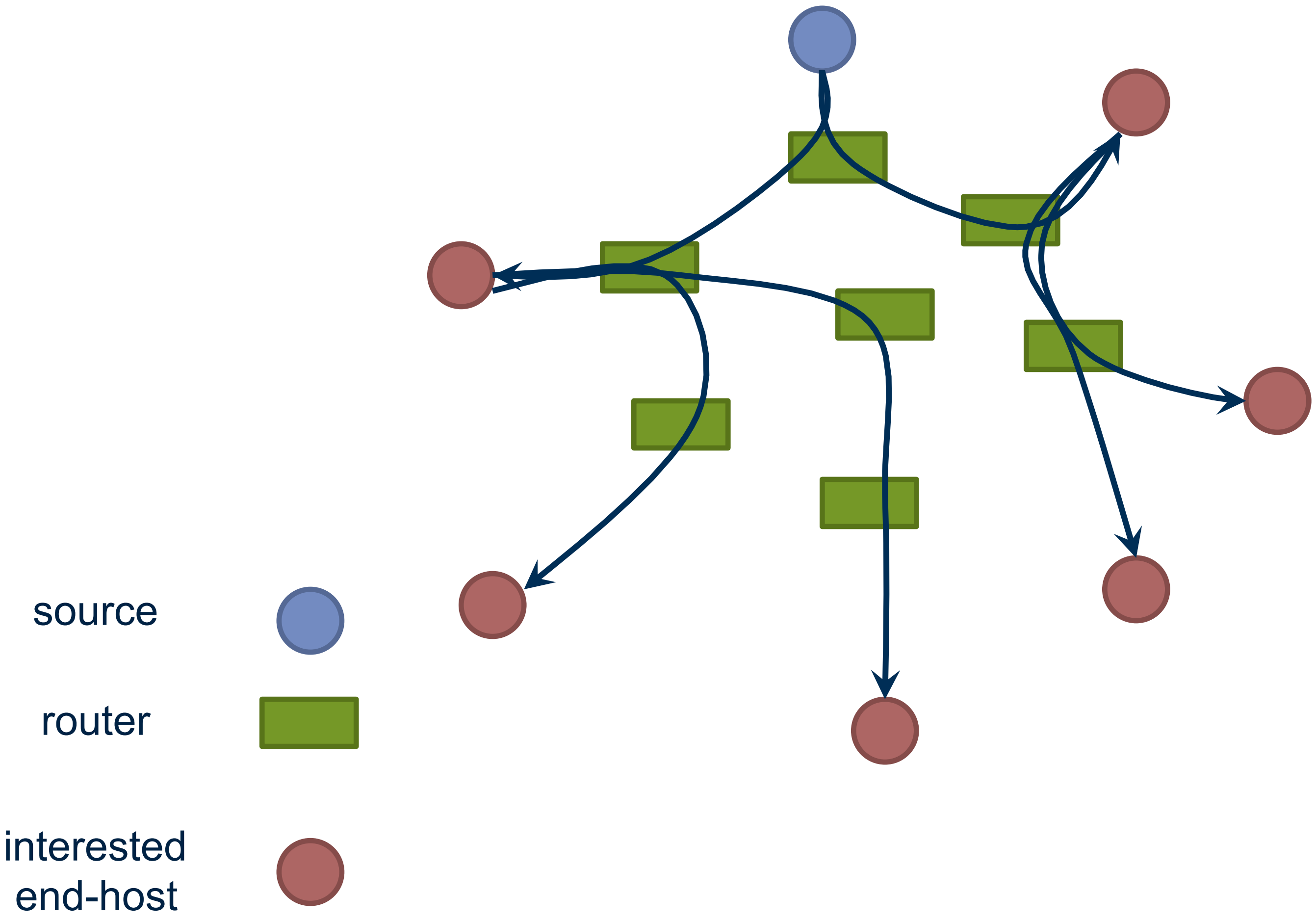


source 🔵

router 🟩

interested end-host 🔴

End-host based multicast



source

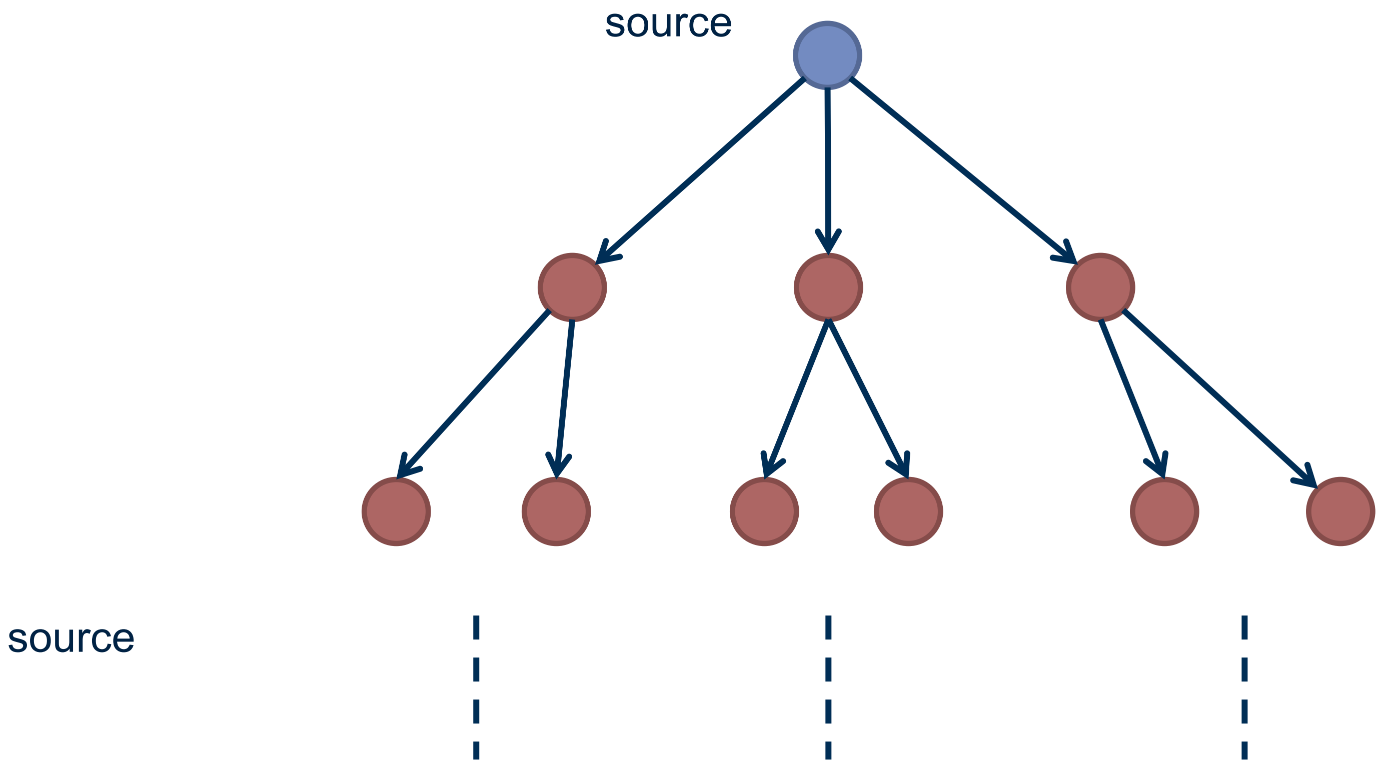
router

interested end-host

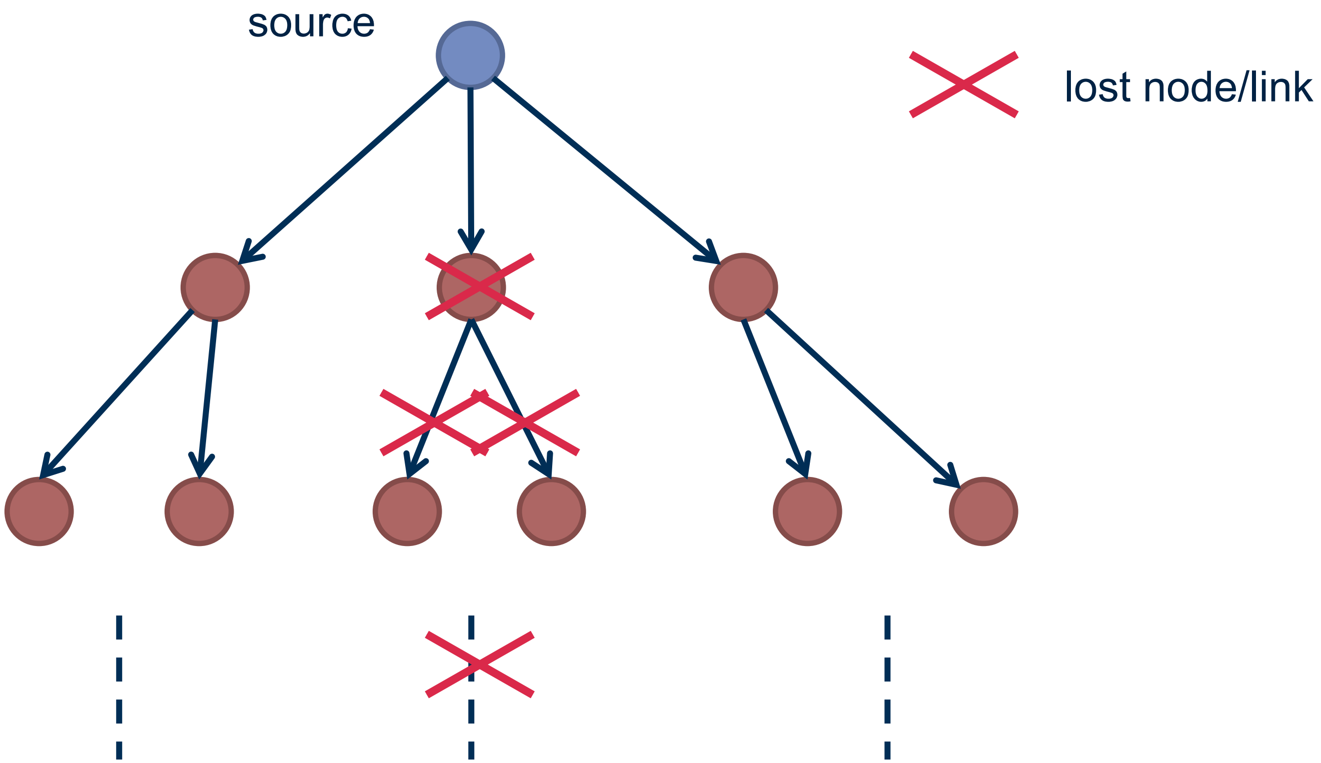
End-host based multicast

- Single uploader → Multiple uploaders
  - Lots of nodes want to download
  - Make use of their uploading abilities as well
  - Node that has downloaded (part of) file will then upload it to other nodes.
- Uploading costs amortised across all nodes
- Also called “Application-level Multicast”
- Many protocols proposed early in the last decade
  - Yoid (2000), Narada (2000), Overcast (2000), ALMI (2001)
    - All use single trees
    - Problem with single trees?

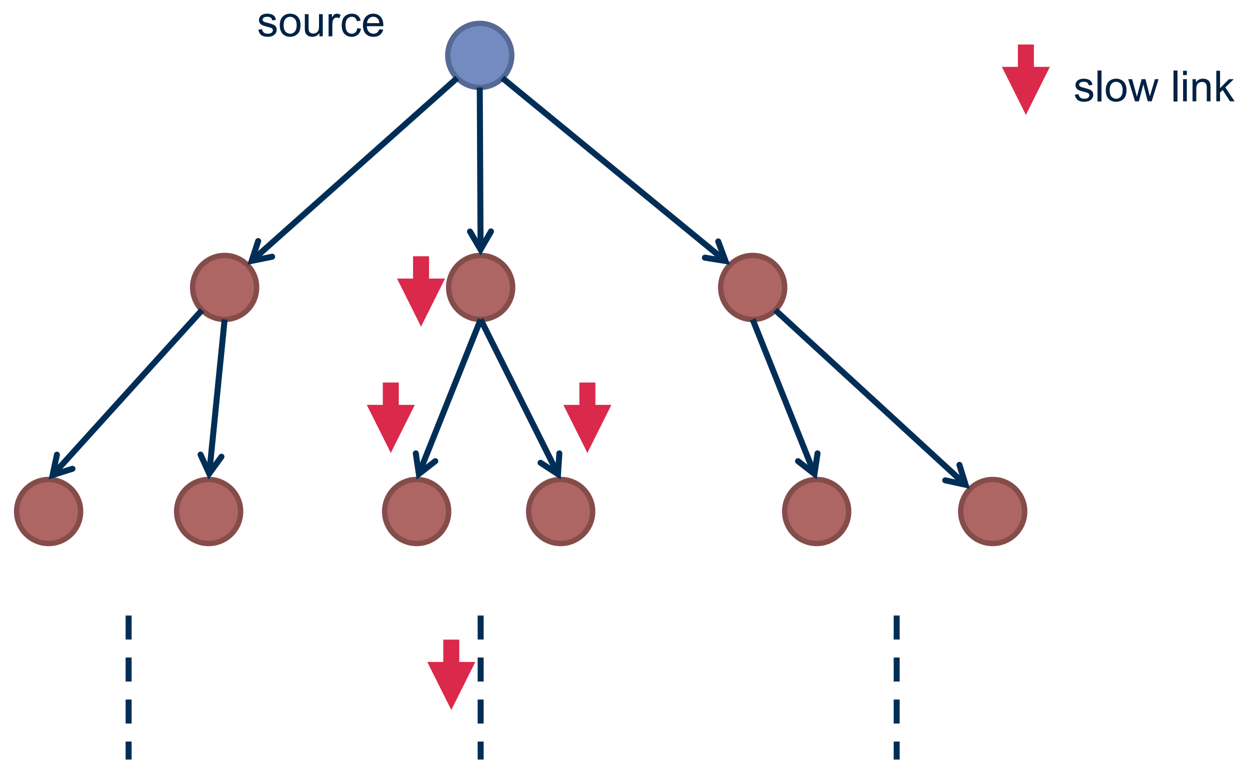
End-host multicast using single tree



End-host multicast using single tree



End-host multicast using single tree



End-host multicast using single tree

- Tree is push-based
  - Node receives data, pushes data to children
  - Failure of interior node affects downloads in entire subtree rooted at node
  - Slow interior node similarly affects entire subtree
- Also, leaf-nodes don’t do any sending

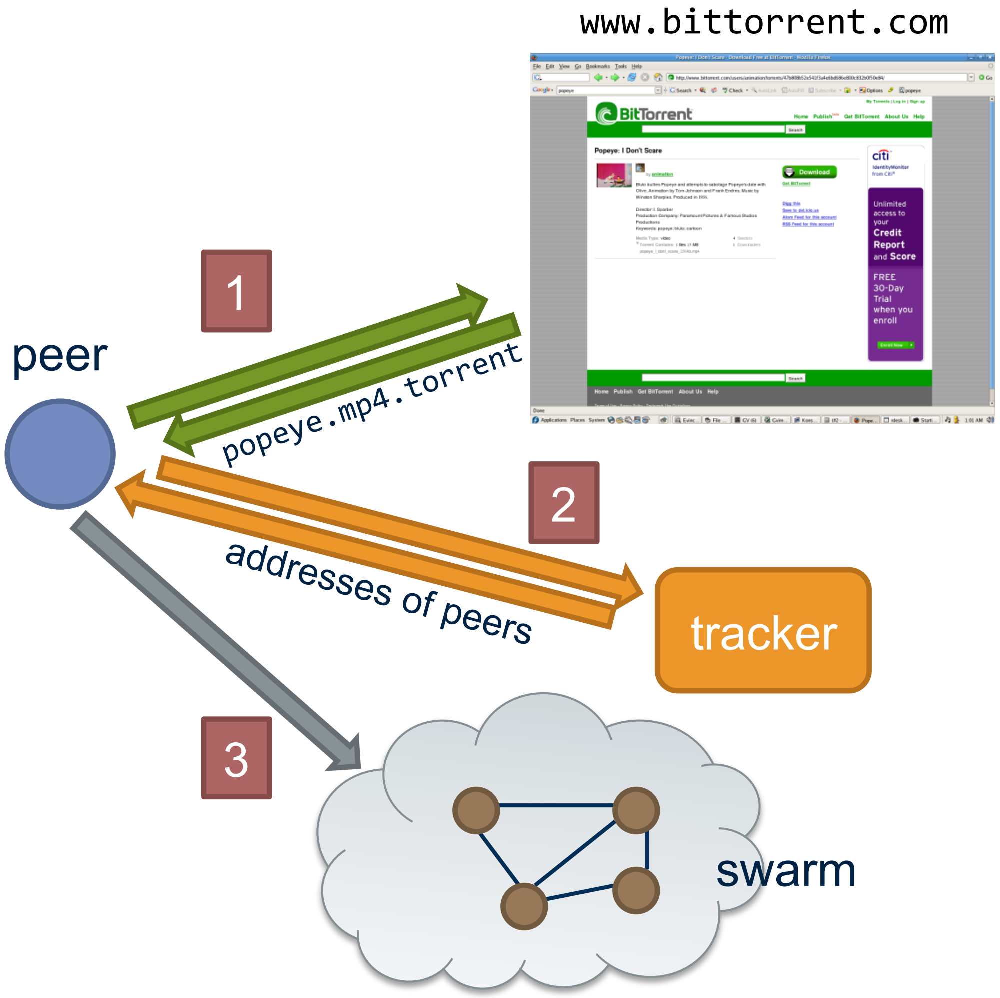
BitTorrent

- Written by Bram Cohen (in Python) in 2001
- Pull-based “swarming” approach
  - Each file split into smaller pieces
  - Nodes request desired pieces from neighbors
    - As opposed to parents pushing data that they receive
  - Pieces not downloaded in sequential order
  - Previous multicast schemes aimed to support streaming; BitTorrent does not
- Encourages contribution by all nodes

BitTorrent swarm

- Swarm
  - Set of peers all downloading the same file
  - Organized as a random mesh
- Each node knows list of pieces downloaded by neighbors
- Node requests pieces it does not own from neighbors
  - Exact method explained later

Entering a swarm for file `popeye.mp4`


www.bittorrent.com
1
peer
popeye.mp4.torrent
2
addresses of peers
tracker
3
swarm
Popeye: I Don't Scare - Download Free at BitTorrent - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
BitTorrent
Home Publish Get BitTorrent About Us Help
Search
Download
Get BitTorrent
citi
Unlimited access to your Credit Report and Score
FREE 30-Day Trial when you enroll

http://www.bittorrent.com/users/animation/torrents/47b808b52e541f3a4e6bd686e800c832b0f50e84/

Popeye: I Don't Scare

by animation

Bluto bullies Popeye and attempts to sabotage Popeye's date with Olive. Animation by Tom Johnson and Frank Endres. Music by Winston Sharples. Produced in 1936.

Director: I. Sparber
Production Company: Paramount Pictures & Famous Studios Productions
Keywords: popeye; bluto; cartoon

Media Type: video
Torrent Contains: 1 files 13 MB
popeye_i_dont_scare_256kb.mp4

- File `popeye.mp4.torrent` hosted at a (well-known) webserver
- The .torrent has address of tracker for file
- The tracker, which runs on a webserver as well, keeps track of all peers downloading file

Contents of `.torrent` file and terminology

- URL of tracker
- Piece length – Usually 256 KB
- SHA-1 hashes of each piece in file
  - For reliability
- Files: allows download of multiple files
- Terminology
  - Seed: peer with the entire file
    - Original Seed: The first seed
  - Leech: peer that's downloading the file
    - Fairer term might have been downloader
  - Sub-piece: Further subdivision of a piece
    - The unit for requests is a sub-piece
    - But a peer uploads only after assembling complete piece

Peer-peer transactions: choosing pieces to request

- Rarest-first:
  - Look at all pieces at all peers, and request piece that's owned by fewest peers
  - Increases diversity in the pieces downloaded
    - Avoids case where a node and each of its peers have exactly the same pieces; increases throughput
  - Increases likelihood all pieces still available even if original seed leaves before any one node has downloaded entire file

- Random First Piece:
  - When peer starts to download, request random piece.
    - So as to assemble first complete piece quickly
    - Then participate in uploads
  - When first complete piece assembled, switch to rarest-first

- End-game mode:
  - When requests sent for all sub-pieces, (re)send requests to all peers.
  - To speed up completion of download
  - Cancel request for downloaded sub-pieces

Tit-for-tat: incentive to upload

- Want to encourage all peers to contribute
- Peer *A* said to choke peer *B* if it (*A*) decides not to upload to *B*
- Each peer (say *A*) unchokes at most 4 interested peers at any time
  - The three with the largest upload rates to *A*
    - Where the tit-for-tat comes in
  - Another randomly chosen (optimistic unchoke)
    - To periodically look for better choices
- A peer is said to be snubbed if each of its peers chokes it
- To handle this, snubbed peer stops uploading to its peers
- Optimistic unchoking done more often
  - Hope that we will discover a new peer that will upload to us

Why BitTorrent took off

- Better performance through pull-based transfer
  - Slow nodes do not bog down other nodes
- Allows uploading from hosts that have downloaded parts of a file
  - In common with other end-host based multicast schemes
- Practical Reasons (perhaps more important!)
  - Working implementation (Bram Cohen) with simple well-defined interfaces for plugging in new content
  - Many recent competitors got sued / shut down
    - Napster, Kazaa
  - Does not do search
    - Users use well-known, trusted sources to locate content
    - Avoids the pollution problem, where garbage is passed off as authentic content

Pros and cons of BitTorrent

- Proficient in utilizing partially downloaded files
- Discourages “freeloading”
  - By rewarding fastest uploaders
- Encourages diversity through “rarest-first”
  - Extends lifetime of swarm
- Works well for popular content

✓

- Assumes all interested peers active at same time; performance deteriorates if swarm “cools off”
- Even worse: no trackers for obscure content

✗

- Dependence on centralized tracker: pro/con?
  - Single point of failure ✗
    - New nodes can’t enter swarm if tracker goes down
  - Lack of a search feature
    - Prevents pollution attacks ✓
    - Users need to resort to out-of-band search: well known torrent-hosting sites / plain old web-search ✗

“Trackerless” BitTorrent

- To be more precise, “BitTorrent without a centralized-tracker”
- E.g.: Azureus
- Uses a Distributed Hash Table (Kademlia DHT)
- Tracker run by a normal end-host (not a web-server anymore)
  - The original seeder could itself be the tracker
  - Or have a node in the DHT randomly picked to act as the tracker

Summary

- Described a large-scale file sharing system
  - BitTorrent
  - Out-of-the-box thinking
- Discussed the salient features of the system
- Described the pros and cons of its design decisions
- The right tool for the job
  - Sometimes, going “extreme” does not require extremely complicated infrastructure
  - But extremely well-executed targeted solutions